Open-Apple™

Releasing the power to everyone.

February 1987
Vol. 3, No. 1

ISSN 0885-4017
newstand price: $2.00
photocopy charge per page: $0.15

Apple introduces an updated IIe

The Apple IIe now features an expanded keyboard with a numeric keypad, a platinum color scheme, and 128K of RAM memory, Apple announced in mid-January. The new model is otherwise little changed from its predecessor — even the price remains the same at $829, suggested retail.

The new keyboard has the same layout as the Apple IIgs keyboard, but the heavier feel of the older IIe keyboard. The reset key has been moved above the ESC and 1 keys and the power light is now above the slash key on the numeric keypad. The numeric keypad's CLEAR key, which returns a control-X on the IIgs, is a duplicate ESC key on the new IIe. By means of a hardware-jumper cut-and-paste operation, you can make it match the IIgs, however.

In addition to the visible changes on the exterior of the computer, the machine uses a revised motherboard that Apple has actually been putting in new IIes since October. The new board includes 64K of RAM in just two chips instead of the previous eight and uses just one ROM chip instead of the previous two. The contents of the ROM are unchanged. The motherboard has also been revised slightly so that an additional 64K of memory can be placed on a very small card that is inserted into the auxiliary slot during manufacturing. Thus, all units now come with 128K of memory. The internal numeric keypad connector was retained.

The shift-key mod. In order to allow mouse-based software to detect a "shift-click" (simultaneous presses on a shift key and the mouse button), the new motherboard comes with the "shift-key modification" already wired. This modification originated with the Apple II-Plus, which dosen't support lower-case characters. The II-Plus shift keys affect only the dual-character number and symbol keys. In order to make the shift keys functional with letters too, a scheme was devised whereby users could wire the shift key to the third pushbutton input on the game connector. By monitoring that "push-button," word processors and other programs can determine whether the II-Plus shift key is being pressed.

When the original Apple IIe was released, jumper pads were provided at motherboard location X6 to make this an easy modification. Now bow-ties are provided to make it easy to disconnect. Problems can arise because devices that actually use the internal game port's third pushbutton input will cause shift-key-modified machines to go into a permanent shifted state unless the modification is removed. In addition, simultaneously pressing the shift key and the third pushbutton input will cause a short circuit on shift-key-modified machines and shut down the Apple IIe power supply.

Since the open-apple and solid-apple/option keys are wired to the game connector's first and second pushbutton inputs, this chart and program show you how to read all three keys:

decimal	-decimal	hex	description
49249	-16287	$C061	PB0, open-apple
49250	-16286	$C062	PB1, solid-option
49251	-16285	$C063	PB2, shift-key mod

```
10 IF PEEK(49249)>127 THEN PRINT "You're pressing the open-apple key."
30 IF PEEK(49250)>127 THEN PRINT "You're pressing the solid-option key."
40 IF PEEK(49251)>127 THEN PRINT "You're pressing the shift key."
50 GOTO 10
```

If this program says you're pressing the shift key when you're not, then the "shift-key mod" is disconnected. This is always the case on the IIgs, which has another means, known as the "modifier-key status register," of telling you not only the status of shift, but also of control, caps lock, option, and open-apple, as well as whether any key on the numeric keypad is down, whether any of the ASCII-code keys is down, or whether the keyboard is repeating. This register is at byte 49189 ($C025). Here's an Applesoft program that shows how to use it, with thanks to a little subroutine explained last month on page 2.92:

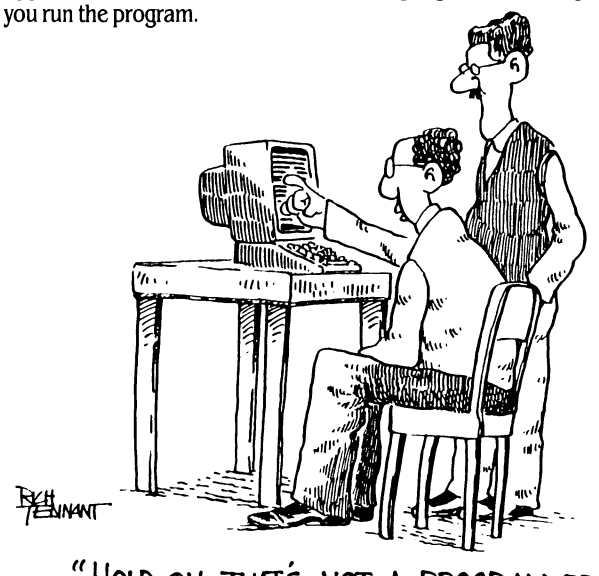
```
10 D=PEEK(49189) : GOSUB 1510 : HOME
20 IF B(7)=1 THEN PRINT "The open-apple key is pressed."
30 IF B(6)=1 THEN PRINT "The option key is pressed."
40 IF B(5)=0 THEN PRINT "An ASCII-code key is pressed."
50 IF B(4)=1 THEN PRINT "A numeric keypad key is pressed."
60 IF B(3)=1 THEN PRINT "The keyboard is repeating."
70 IF B(2)=1 THEN PRINT "The caps lock key is pressed."
80 IF B(1)=1 THEN PRINT "The control key is pressed."
90 IF B(0)=1 THEN PRINT "A shift key is pressed."
99 POKE 49168,0 : GOTO 10 : REM clear keyboard strobe

1510 FOR N=7 TO 0 STEP -1
1520 : BN=2^N : B(N)=0 : IF D>BN-1 THEN D=D-BN : B(N)=1
1530 NEXT : RETURN
```

The reason a program might want to know if a numeric keypad key is being pressed is so that it can tell between, for example, the keypad's slash key and the main keyboard's slash key. All of the keypad's keys have main keyboard duplicates. Also notice that bit 5 is meaningful when it has been cleared to zero, while the others are meaningful when they're set to one.

Maybe even more exciting than a new IIe to many of you is Beagle Bros' new ProDOS-based Applesoft compiler. The *Beagle Compiler*, by Alan Bird, is the first ProDOS-based Applesoft compiler we've come across ($74.95, 3990 Old Town Ave, San Diego CA 92110 619-296-6400).

A compiler does to your Applesoft program what a human translator does to documents written in languages you don't understand. As far as your Apple is concerned, a program written in Applesoft is in a foreign language. The program has to be translated before your Apple can execute it. Normally, Applesoft itself does the translation *as the program is running, every time* you run the program.



"HOLD ON, THAT'S NOT A PROGRAM ERROR, IT'S JUST A BOOGER ON THE SCREEN."

A compiler translates an Applesoft program into something your Apple can understand one time. No further translation is necessary. Consequently, the program will execute faster. How much faster depends on the program. String-intensive programs can run as much as 5 to 15 times faster than normal (compare this to the 2.5 to 3 times boost you get from a hardware accelerator). Programs that consist primarily of floating point calculations, on the other hand, don't run much faster at all.

In addition to being ProDOS-based, Beagle's Applesoft compiler has several other important features—compilation is fast, compilation is simple, and compiled programs are *shorter* than the original. Ampersand routines without parameters will compile without problems and Beagle's manual explains how to modify ampersand routines that do pass parameters. The *Beagle Compiler* works with any 64K or larger Apple II. It's an order of magnitude better than any other Applesoft compiler I've ever seen.

Apple released an updated ProDOS system disk to developers in mid-January. It contains version 1.1 of ProDOS 16 and version 1.3 of ProDOS 8. Both are dated 2-DEC-86. Among other changes, the new versions fix the bug in the ProDOS floppy driver discussed here in October.

Apple's new SCSI hard disk controller card has some problems running on the IIgs. Apple is recommending that users back up their disk often while its engineers work out the bug. Bob Shofstall at Nite Owl Productions, who has a SCSI drive on a IIgs, says he can hardly keep it running long enough to back it up.

Apple has also discovered that at least one revision of its DuoDisk has one too many resistors to work correctly on a IIgs Smartport daisy chain. Tell your dealer to look on AppleLink for more information on this one if you're having problems—the DuoDisk can be made to work by cutting the extra resistor off the DuoDisk's internal logic board.

Two of the most frequently asked questions about the IIgs at the moment are how to wire older Disk II-type drives so they can be connected to the IIgs Smartport daisy chain and how to wire non-Apple analog RGB monitors to the IIgs. We printed a diagram showing how to convert the 20-pin "header" connector used by older drives to the newer DB-19 connector in December 1985, page 90. It's all very straightforward—the only twist is in the drive enable lines. Pin 17 on the DB-19 carries the drive enable signal for drive 1, pin 9 carries the signal for drive 2. In each case the drive expects its own enable signal on pin 14 of the header connector.

I know far less about RGB cables, but here, at least, is the pinout of the IIgs video connector:

```
    8  7  6  5  4  3  2  1   pin layout of IIgs female video connector
     15 14 13 12 11 10  9    from connector side (not solder side)
```

1	ground	ground reference and supply
2	red	red analog video signal
3	comp	composite synch signal
4	N.C.	no connection
5	green	green analog video signal
6	ground	ground reference and supply
7	N.C.	no connection
8	+12V	+12 volt supply
9	blue	blue analog video signal
10	N.C.	no connection
11	sound	analog sound output
12	NTSC/PAL	composite video output
13	ground	ground reference and supply
14	N.C.	no connection
15	N.C.	no connection

Uncle DOS is anxious to hear from anyone who can shed more light on what kinds of RGB monitors will work with the IIgs and how they should be connected.

***Apple Writer* guru Don Lancaster** has already come up with a patch that allows *Apple Writer* to work on the IIgs. As mentioned last month, ProDOS-based *Apple Writer* won't print through the IIgs serial ports. Lancaster's patches make *Apple Writer* realize that what appears to be a Super Serial Card in the IIgs isn't really a Super Serial Card at all. After the patches, *Apple Writer* treats the IIgs as if it had a third-party interface card (and treats Super Serial Cards themselves as if they were third-party cards, too). This means, among other things, that the patch must also disable the control-O J "set modem/printer interface" option, and it does. The patch is for 128K ProDOS *Apple Writer* version 2.0 only:

```
   from the Monitor, enter
BLOAD AWD.SYS,A$2000,TSYS

   verify that 4DB0 is A0, then
4DB0:60
   verify that 4F67 is 01, then
4F67:10
   verify that 4F6E is 31, then
4F6E:13

BSAVE AWD.SYS,A$2000,TSYS
```

Back in November I mentioned (page 2.77) that I had never been able to get *Apple Writer's* "peek" command, which allows you to see what's in a file without actually loading it, to work. The manual says you can do this by acting as if you're going to (L)oad the file, but then adding a backslash after the file's name. Every time I tried this I got the message "ProDOS error: Bad Pathname."

Subscribers Robert Hall and Robert Flickenger have come to my rescue. It turns out that what the manual really means is that you must append whatever character you are using for an underline token onto the end of the filename. The underline token appears on the (P)rint menu. Since I don't use *Apple Writer's* underline token, I had it defined as nothing at all. No wonder I couldn't get the file peeker to work.

***The Wall Street Journal* is by far my favorite newspaper** ($119/yr, 200 Burnett Road, Chicopee, MA 01020 800-841-8000). It's far more than a financial journal. For example, look at this story, which appeared on December 30, 1986:

When John Sculley, chief executive of Apple Computer Inc., addressed the First Boston conference on high technology earlier this month, he said Apple had recently bought a Cray Research Inc. supercomputer for about $14.5 million and was using it to develop its next-generation Apple.

When it came time for John Rollwagen, Cray's chief executive, to address the conference, Mr. Rollwagen said he hadn't wanted Apple to think this was a one-way street. So, he said, "since they were good enough to buy one of our machines, some of us have bought a few of theirs."

Mr. Rollwagen also said he told Seymour Cray, the company founder, about how Apple was using the machine. "There was a pause on the other end of the line," Mr. Rollwagen recalls, "and Seymour said, 'That's interesting, because I'm designing the next Cray with an Apple.'"

Another publication I like a lot is *Science News* ($29.50/yr, 231 W Center St, Marion, OH 43306 800-247-2160). The best feature of this magazine is that it limits itself to 16 pages, front and back covers included, week after week. From the December 20, 1986 issue (page 391) comes news of an experiment by Diana Deutsch of the University of California at San Diego that concerns pairs of tones, constructed from a set of sinusoidal waves, that are a half octave apart. Some listeners hear the second tone as higher in pitch than the first, while other listeners hear the second tone as lower in pitch. Furthermore, if the tones are taped and played back at different speeds, a listener who hears the pattern as ascending at one speed may hear it as descending at another. Yet, for any single individual, a given pattern and speed will clearly and consistently sound either ascending or descending.

"In visual terms," *Science News* says, "this musical effect is as paradoxical as seeing a square instead of a circle when a shape is shifted to a different location." (Or as paradoxical as one person seeing a square where another sees a circle in the same location, I might add.) The effect would be important to any of you trying to use sound as a data-analysis tool, as suggested here in March and September 1986 (pages 2.9 and 2.58). Apparently the effect is limited to certain types of tones. "The experiments don't work on, say, a piano keyboard, because each piano note has a more complex structure than the experimental tones (which sound more like an organ) and listeners apparently use those extra clues to determine relative pitch." I suspect there are some amazing discoveries yet to made in the area of sound and I suspect the Apple IIgs is the perfect research ship to make them in.

Those of you interested in learning more about writing Applesoft programs that use ProDOS should take a look at the new book *ProDOS Inside and Out*, by ***Open-Apple*** technical consultant Dennis Doms. Dennis borrowed a few chapters from my *Softalk* and ***Open-Apple*** articles (that's how I got my name on the cover, too), but most of it is new stuff. Unlike most other books you'll find on ProDOS, *ProDOS Inside and Out* deals almost exclusively with Basic.system (in fact, Dennis and I wanted the book to be

called *The Basic.system of Apple's ProDOS*, but changing the mind of a publishing company can be difficult).

The book has five major parts, which consist of six introductory chapters, five chapters on Applesoft programming, five chapters on programming with text files, five chapters on assembly language programming, and four chapters on using the ProDOS machine language interface. The book is available (or can be ordered) from your favorite bookstore (Tab Books is the publisher), or you can get it from us at the suggested retail price of $16.95, postage paid. We also have a disk with the book's 40-plus programs for $10, which your bookstore will never have.

***Open-Apple: Vol 2*—a bound, indexed edition** of our February 1986 through January 1987 issues, is also now available, directly from us, for $14.95. Book orders are handled fastest if you send them to our Overland Park address (see back page) *unless* they also have something to do with your subscription (payment, change of address, missing issue, etc.), in which case it is *always* fastest to use our Syracuse address.

Open-Apple* has a new apologist and business manager, Sally Dwyer.** Among her many jobs is to make sure all of our customers have been satisfied with our service. Unlike our circulation manager Sally Tally, who is an imaginary being whose chores are done by a number of people (none of whom are actually employees of ***Open-Apple), Sally Dwyer is a real person with the patience of a first-grade teacher. If you every have problems with your subscription or with a product order, contact Sally at our Overland Park address and she'll see that you're satisfied.

(A note to those of you who got an invoice or renewal slip this month—please don't complain to Sally that you didn't get an envelope to return your payment in. Enclosing both the index for last year and a return envelope would have made the newsletter too heavy to send without more postage. Return envelopes will return next month.)

Do we have any *Stats Pro* users out there? Subscriber Jean McCollom would like to trade tips on using the program with you (this is how special interest groups get started). You can reach McCollom at 813-657-2531.

Ask (or tell) Uncle DOS

I've always thought that writing assembly language programs was one of the best ways available to develop the virtue of humility. That's because no matter how hard you try nor how long you work there's always at least one more bug. Now I think writing a newsletter about computers may be even better for humility than programming. Look, I'm bursting out all over with humility. Grab your pencils and your back issues, folks.

This month I'll start with a correction to last month's corrections. On page 2.93, the line that begins with 382 should say POKE 49152 instead of 49236.

Then, on page 2.96, take those dollar signs out of the BSAVE FID line in the first column—those are decimal numbers.

*Back in October, on page 2.71, I said the mnemonic meaning of the program FISHEAD had been lost. Rich Williams, the Apple programmer who wrote FISHEAD and named it, collared me recently and said that the meaning isn't lost at all—it stands for **FI**le **S**hower, **HE**lper, **A**nd **D**uplicator. Next time I see him he'll probably tell me I spelled it wrong, but I'm not touching this fish story again.*

And that's not all folks...

Smartport lines missing

It appears that the dastardly and devilish denizen of DOS deliberations has done diddled with Uncle DOS's disk drive device disassemblies. On the very same page that you corrected the AUXMEM program bugs (January 1987, page 2.93), there appears to be something lacking in the "Known Bugs" paragraph (how apropos, a bug in the bug finder). First, in line 680 shouldn't it be POKE 817,52, not POKE 817,53? This is needed to agree with the PEEK(820) command. Secondly, after entering all those POKEs in lines 660 to 681, I could find nowhere in the code where the addresses were CALLed. Is perhaps a line such as "682 CALL 809" missing?

Anyway, those are the changes that I made and the program tells me that my UniDisk 3.5 card is a baddy. If this is so, what can I do about it? I always write-protect disks that I don't want to lose, and I absolutely refuse to stop doing this. What a dilemma!

Craig Peterson
Santa Monica, Calif.

You are correct. Even though we tested that program extensively with line 680's POKE 817,52 fixed and a CALL 809 at line 682, those little fixes didn't get into the file we actually typeset.

You can still write-protect disks if you want to. Just don't attempt to save anything important on one of them. If you do, ProDOS will get quite confused and you might lose what you're working on. Often that isn't as important as losing the whole disk, however. It still makes sense to write-protect originals when making copies of disks, for example, so that you don't initialize the original.

Smartport follow-ups

I thoroughly enjoyed your article on Smartport in the January 1987 ***Open Apple.*** With regard to the number of devices that may be connected to the Smartport bus, Apple has set a limitation of four devices due to power supply and signal degradation effects. It may be possible to extend this limitation by designing Smartport devices that supply their own power and buffer all signals on the bus, but this hasn't been done yet.

In addition to the limited capacity of the bus, the ProDOS entry point on Smartport protocol converters also has a maximum capacity of four devices. ProDOS calls to slot 5, drive 1 always go to bus unit 1; slot 5, drive 2 to unit 2; slot 2, drive 1 to unit 3; and slot 2, drive 2 to unit 4.

ProDOS is being updated to use the Smartport directly, however, rather than using the Smartport's ProDOS entry point. The new version will not have the four-device limit. An updated ProDOS, combined with self-powered and buffered bus-devices, would allow full use of the Smartport's capabilities.

Ray Montagne
Smartport Software Engineer
Apple Computer, Inc.

Reading your Smartport discussion was worse than eating salted peanuts—I had to have more. Some details cried out for attention. What, for example, does ProDOS do with a Smartport during the boot process? And how are device numbers assigned? Is there any way to detect a RAMdisk in the chain short of asking every device for its name? Finally, since ProDOS 1.2 knows enough to remap device numbers when more than two devices are found in the Smartport chain, you assertion that "ProDOS doesn't know a Smartport from a Dumbport" seemed limited to the world of ProDOS 1.1.1. Just what does ProDOS 1.2 do during system initialization?

With such questions in mind, I disassembled the ProDOS 1.2 initialization code and spent an evening going over it. What I found may be of interest to your readers. (Side note: the IIgs disassembler is of course more powerful than previous Apple II disassemblers, but a curious problem crept in: Since the 65816 has 256 legal opcodes, the venerable "???" has all but disappeared from disassemblies; hence a small data cache tacked onto the end of a code module is much harder to spot. The ProDOS title page text, for example, disassembles as legal code.)

As with ProDOS 1.1.1, ProDOS 1.2 does a slot scan as it builds the system global page. Immediately prior to beginning the scan, however, ProDOS 1.2 peeks at slot 2. The result is stashed away for later use, and, beginning with slot 7, the slots are checked in descending order for the presence of block devices. If ProDOS 1.2 finds a block device which is not a Disk II, however, it specifically checks for the presence of Smartport protocol converter firmware by testing the ID byte at $Cn07. If a Smartport is found, then its entry point is determined and a status call is made via a cute little piece of self-modifying code. The status call gives ProDOS the number of devices accessible through the Smartport and the first two of these devices (if extant) are assigned device numbers corresponding to the slot in which the Smartport is found.

Then an interesting thing happens—unless the Smartport is found in slot 5, ProDOS ignores any other devices in the chain. If the Smartport is indeed in slot 5, then the slot 2 status previously stashed away is exhumed and checked; if a block device exists in slot 2, then ProDOS washes its hands of slot 5. Otherwise, the third and fourth devices in slot 5 (if extant) are given device numbers corresponding to slot 2 with device driver addresses in slot 5. ProDOS then moves on to slot 4.

So what does all this mean? As I see it, there are several conclusions to be drawn:

1. ProDOS 1.2 will recognize up to four devices controlled by a Smartport in slot 5, where the IIgs Smartport lives. Extra devices in other slots are ignored. This is in agreement with the claim made in the *IIgs Firmware Reference Manual*. As far as I can tell, however, ProDOS 1.2 does not discriminate here in favor of the IIgs. Thus, connecting more than two devices to a Smartport in an older Apple II appears to be beneficial only as long as the Smartport resides in slot 5.

2. If for some reason a block device exists in slot 2 (an extra memory expansion card, perhaps?) then ProDOS 1.2 will ignore anything beyond the first two devices attached to a Smartport in slot 5.

3. There doesn't appear to be a simple method of detecting a RAMdisk attached to the IIgs Smartport short of quizzing the Smartport yourself. ProDOS knows only the number of devices attached to the Smartport at initialization, and assigns device numbers accordingly. Thus, while an Apple memory expansion card is readily identified by the ID byte at $CnFB, a RAMdisk attached to the Smartport is just another anonymous block device. I suspect this is the reason that Catalyst 3.0 doesn't recognize the IIgs RAMdisk as such. Do you or your readers know of a cheap way to recognize the presence of a IIgs RAMdisk?

Finally, a few observations about the IIgs in general. I've been using a IIgs steadily since the end of November, and have concluded that a) it's an impressive machine (of course), b) the memory expansion card is a virtual necessity, and c) the genius who decided to attach the mouse to the keyboard apparently never twigged to the fact that detached keyboards tend to reside on laps rather than desktops. I am constantly detaching and re-attaching the mouse as I shift between text- and graphics-oriented applications.

Paul Harwood
NoetiComp, Inc.
West St. Paul, Minn.

I don't know of any other way to identify a RAMdisk that's in a Smartport chain than to do the calls outlined here last month.

The problem you mention with no "???s" in IIgs disassemblies is alleviated somewhat by the fact that the IIgs memory dump includes an ASCII display. Try BLOADing ProDOS 1.2 at $2000, entering the Monitor, and typing 2000.5D00. You'll find all kinds of interesting text embedded in ProDOS 1.2.

I totally agree about the necessity of adding a RAM card to the IIgs. The bare 256K machine is fine for older programs, but precious few ProDOS 16 programs will run with that little memory.

Escape route flaw fixed

I tried the technique for embedding ESC sequences in AppleWorks files described in your January 1987 issue (page 2.95). I was able to modify color, density, and other printer features from text as described. However, the initial letters following carriage returns were deleted.

The problem seems to be that AppleWorks sends the current boldface command at the beginning of each line. If that command is a simple ESC, the printer takes the first letters of the line as part of a printer command and does not print them. The printer sometimes even begins to do strange things such as reverse linefeeds, erratic color changes, and strange character sizes.

My solution is to leave the "bold off" control sequence unchanged. I use control-B(oldface) to send ESC, type in the printer command I want, then use control-B again to make sure the boldface-off code, not the boldface-on code, will be sent at the beginning of the next line.

Anna Martin
Houston, TX

AppleWorks vs chemistry

In the November, 1986 issue you claim that AppleWorks will do 50 per cent of what advanced users want to do. Unfortunately, I am in the other 50 per cent. The reason is that my Apple spends much of its time writing materials for my chemistry courses—tests, quizzes, etc. To do this I need to use a lot of special printer features—in addition to underlining, subscripts, and superscripts I use alternate user-defined character sets (on an Epson FX-80), expanded type, and italics. I realize that I can have one of these last three by redefining the boldface command, but I need all three. (At present I'm using a version of Screenwriter II that I patched up myself to allow access to these and other printer features).

I can see, based on a little experience, that AppleWorks has a lot of advantages and I would certainly like to shift over to using it. Hasn't anyone tackled the problem of fixing this most serious and obvious defect of AppleWorks, and providing a way to enter arbitrary printer command strings into AppleWorks data files?

It would also be nice to be able to get into, and out of, the various printer modes more easily; in chemistry one might use as many as twenty sub- and superscripts in a single line, which gets to be a lot of keystrokes. Maybe one of the utilities that allow macro definitions would help here.

Walter K. Dean
Huntington Woods, Mich.

We get more letters about adding printer functions to AppleWorks than on any other subject. And we've printed quite a few of those letters here in ***Open-Apple****. Now might be a good time to summarize and bring together all we've learned about this subject so far.*

First, there is no way to embed control characters within an AppleWorks document. AppleWorks itself embeds control characters into word processor files and uses them for its own purposes. Even if you managed to slip a control character into an AppleWorks file using a disk zap utility, AppleWorks would not send it to your printer.

This leaves you with the built-in AppleWorks printer controls. AppleWorks allows you to define three printers, only one of which may be a "custom printer." The limit of just one custom printer is hard to live with. One trick that makes it somewhat easier is to use the "Silentype" printer, rather than the custom printer, for printing formatted text files to disk. See ***Open-Apple****, August 1985, page 60b, for more information on formatted text files. Since the Silentype doesn't support much of anything in the way of printer features, no control-codes will be appear in files printed to disk—which is what most people want, anyhow.*

So now you can use your custom printer definition to tell AppleWorks lies about your printer's control codes; this allows you to get your printer to do italics, for example, when AppleWorks thinks it's doing boldface. AppleWorks stores the printer codes you define in a file called SEG.PR. Because SEG.PR isn't kept in memory but is always loaded right before printing a document, you can effectively increase the number of custom printers by making several copies of SEG.PR with different control codes and switching disks when AppleWorks isn't looking.

Custom printer controls supported by AppleWorks include 21 settings for characters-per-inch, two settings for lines-per-inch, and four bold, underlining, superscript, and subscript on-off pairs. The four on-off pairs can be used within lines, but the per-inch codes can be used only between paragraphs. For printer features that you want to switch on and off for whole documents (such as printing a document in near-letter-quality), rather than within a document, two other possibilities are to add the necessary control characters to a printer's interface-card setup string, or to print an otherwise blank spreadsheet after specifying the control characters you need with open-apple-O's "Send Special Codes to printer."

As mentioned, the characters-per-inch and lines-per-inch settings can only be changed between paragraphs. To use the per-inch controls effectively, you have to turn on the fake per-inch (say you have your printer's code for italics stored in the 24 characters-per-inch control), print a sticky space and a return, then turn the per-inch setting your printer is really using back on. If you don't go back to the correct setting, AppleWorks will put too many or too few characters on each line or lines on each page.

The boldface, underline, superscript, and subscript on-off pairs can be used within a line or paragraph. The limit most people run up against is that, like you, they have five or more features they want to turn on and off. Before giving up completely, one last trick is to define the on-code of one of the pairs as simply an escape character (see Anna Martin's letter above for more detail on this trick). Your printer, the FX-80, uses ESC 4 and ESC 5 to turn italics on and off, ESC E and ESC F to turn emphasized print on and off, and ESC G and ESC H to turn double strike on and off. If you define boldface-on as simply ESC, try this (the carets are bold on and off codes):

```
^4^Italics^5^ aren't ^E^^G^bold^F^^H^.
```

You should get something like this:

Italics aren't **bold**.

The only remaining problem with this trick is that AppleWorks will count the 4, 5, E, G, F, and H as characters when deciding how much to put on one line, but your printer won't actually print any of those characters. Consequently, lines with lots of codes will be "wrapped" before they should be.

One of the AppleWorks macro programs would, of course, make entry of these codes much easier.

One very nice feature that AppleWorks provides the Imagewriter that it doesn't provide "custom" printers is support of justified proportional type. To get both proportional type and the ESC trick we've just presented, see the next letter.

If combinations of these tricks still don't provide you the printer control you want, it's time to give up on AppleWorks. If AppleWorks could do everything, after all, there wouldn't be any need for the many other word processors that are available for the Apple II.

You might also consider a program, such as ***FontWorks*** *from the Software Touch, that allows you to add printer embellishments to files created with AppleWorks or with other word processors.*

Zapping Imagewriter codes

I've become (by default) the Apple answer man at the high school where I teach math. The November newsletter item about printing a whole AppleWorks document in bold was of interest because the Imagewriter needs the double strike to make a good ditto master. Actually, since I use AppleWriter, I had no trouble doing so but many of my colleagues use AppleWorks. I've combined Dean Shutt's tip with one that appeared in the December issue of *A+* as follows.

Use a disk zap program (I use *Bag of Tricks 2*) to search the AppleWorks program file SEG.PR for the

hex string 02 1B 21 02 1B 22. The $1B $21 and $1B $22 are the ImageWriter codes for bold begin (ESC !) and bold end (ESC "). The $02's seem to tell AppleWorks how many characters are in the code sequence. Change both $02's to $01's and write the change to disk. This change causes AppleWorks to send just the ESC character when you embed, via a control-B, a bold begin or bold end in your document.

This by itself does nothing. However, as was pointed out in the *A+* tip, if you follow the ESC with a valid Imagewriter control code, it is executed and the characters are not printed in the document. So, to send a bold begin, (ESC !) you would type "control-B !". Unless you turn off bold with ESC ", the whole document is printed in bold. More importantly, though, now you have the *Apple Writer*-like ability to send the sequences necessary for using color, MouseText, half-height characters, and so on while not giving up anything available from the open-apple-O printer options menu.

Howard Jackson
Albany, N.Y.

Note Anna Martin's letter above. Your trick will work more reliably if you leave the bold-off code's length at two and change it to something that's meaningful to your printer but that doesn't turn off something you want on (bold in your case)—for example, try using "ESC f", the code for forward linefeeds. The trick also still has the problem with short lines mentioned in the previous letter.

But what's neat about it is that it doesn't use up the custom printer definition and it allows the ESC-code trick to work with proportional type. Using the same basic disk-zap trick, you could also do such things as embed commands in an AppleWorks printer definition that AppleWorks won't normally let you enter (such as a caret). Somebody might even be able to find the text that makes up the open-apple-O menu in the word processor and change the words "Boldface Begin" and "Boldface End" to "ESC start" and "ESC end."

More disk recovery tricks

In the September 1986 ***Open-Apple*** you mentioned POKE 47246,24 during a discussion of methods for recovering a crashed disk (page 2.59). This POKE changes a set carry instruction inside the DOS 3.3 RWTS routines to a clear carry instruction. At this point the carry bit is being used to tell RWTS whether there was an error in the targeted sector's "address trailer" or "data-checksum trailer". Your POKE makes RWTS ignore these errors.

The following patches take more time but have a better chance of reading bad disks:

```
B989:EA EA      Ignore address checksum

B992:EA EA      Ignore address trailer
B99C:EA EA

B92D:EA EA      Ignore data checksum

B8DF:EA EA      Ignore long gap between address
                   and data
B936:EA EA      Ignore data trailer
B93E:A9 00

B94D:EA EA      Ignore ''can't find address''
                   (keep trying)
For really hard cases only:

B960:EA EA      One byte address header
B96B:EA EA

B8F2:EA EA      One byte data header
B8FD:EA EA
```

These patches should cope with most disk errors and allow them to be read. However the data is probably corrupted and if this happens in a track/sector list the problem is not gone yet. I hope this helps save wasted hours of re-typing.

Charles H. Putney
Dublin, Ireland

POKE 47246,24 handles a fairly common problem caused by a slow disk drive. This is that the last few bytes of a sector are overwritten by the first few bytes of the next sector when the next sector is written onto the disk.

More severe problems can be attacked with your additional patches, but everyone should realize, as you point out, that the more severely RWTS is patched to ignore errors the more likely that any data you read will be corrupted. If you patch RWTS to completely ignore errors, you have to suspect the integrity of everything and you may get stuck inside an infinite loop trying to read an unreadable sector.

Those who make severe modifications to RWTS should do so with a copy of ***Beneath Apple DOS*** *at hand. Use it to try to understand what exactly these patches do. Any error-checking you remove from DOS you will have to account for yourself.*

Protection errors

Why can't my disk sector editors always read track 0, sector 0? It seems to be unable to deal with disks that have funny sync bytes. Obviously the Disk II controller can read it. What's different?

Jeff Root
Lancaster, Calif.

Your sector editor is using DOS 3.3's RWTS routines to attempt to read the sector. RWTS expects to find some byte patterns that mark the beginning and the end of the "header" and "data" fields for each sector. It also does some other error checking.

The disk controller card, on the other hand, doesn't use RWTS. The routines it uses to read track 0, sector 0 are right there on the controller card. Because of limited space, the program on the controller card has far fewer abilities and does less error checking than RWTS itself. People who do copy protection take advantage of this by introducing errors that stop RWTS but that don't stop the disk controller.

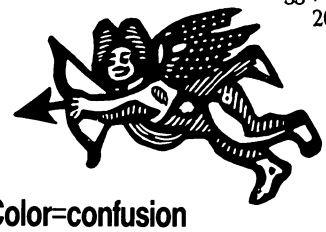
Try judicious use of a few of the POKEs mentioned in the previous letter. I'm sure you'll be able to read the sector you want to read by eliminating the right error. This is why copy-protection is such a waste of time—the computer has to be able to read the copy-protected disk and if the computer can do it so can any human who wants to bad enough. If this topic interests you I suggest you get a subscription to ***Computist****, PO Box 110846-T, Tacoma, WA 98411 ($32/yr).*

General Manager resupported

In regard to the letter in your November issue (page 2.77) about the *General Manager* data base program, I'd like your subscribers to know that my company has taken over marketing and support of the program from Sierra Online and from PC Manager, owners of the program.

I am upgrading the *General Manager* and will be releasing version 2.1A in January 1987. There are still both hard and floppy disk DOS 3.3 versions of the program. Upon reaching my office, any individual receives technical support at no charge. I also write custom applications for the *General Manager* for individuals and organizations. Upgrades are $25.00 for the system disk and the cost of a new program has been reduced to only $179.95.

David Spooner
Aspen Data Systems
Box 567
Driggs, Idaho 83422
208-354-8185



Color=confusion

What causes Applesoft memory to get scrambled occasionally and give me garbage such as "COLOR=" when I try to LIST a program?

Link Keur
Edison, N.J.

Memory can get scrambled any number of ways. "COLOR=" and similar garbage appears when Applesoft thinks there's a Basic program in a certain range of memory when there really isn't. For a lot more on this, see "Applesoft meets the RAMdisk" in our April 1986 issue, pages 2.17-21.

Pascal RAMdisk loader

In the December 1986 ***Open-Apple***, page 2.87, there's a letter and answer about moving large files into a RAM disk on startup. You wrote that you didn't know how to automatically load large files under the Apple Pascal operating system. I am writing to let you know how to do it.

The Pascal Filer will copy files of any length from a floppy disk to a RAM disk. The trick is to get this done automatically on startup. Here are the steps to take:

1. Make an Exec file that invokes the Filer and transfers the Filer from disk to RAM disk. This file is best saved to the boot disk. Type "M" for "Make Exec" and answer the prompt "New Exec Name:" with "*:st" to make the Exec file ST.TEXT on the boot disk. Answer the prompt "Terminator = %, change it?" with "N". When the command line returns, type "f" (for Filer), "t" (for transfer) and at the prompt "Transfer what file?" type the volume and filename (including suffix) of the file to be transferred. At the prompt "To where?" type the volume name of the RAM disk followed by the file name (or "$" if you want to keep the same name). Then type "q" to quit the Filer. Type "%%" to end the Exec file and save it. Check the boot disk directory; ST.TEXT should be there as a 2 block text file.
2. An Exec file cannot itself be a startup file. In order to execute ST.TEXT automatically on startup, a special program called SYSTEM STARTUP must be written which will chain to it. Here is such a program:

```
program sysstart;

  uses
    chainstuff;              { needed to chain }

  begin
    setchain ('exec/*:st')  { chain to (exec) ST }

  end.
```

Save this file as *:SYSTART.TEXT and compile it as "*:$" to keep the same name. Upon successful compilation, use the filer to change SYSSTART.CODE to SYSTEM.STARTUP.

That's it. When the computer is booted using the boot disk, the program SYSTEM.STARTUP will run the

Exec file ST and copy the file or files you specified from the floppy to the RAM disk.

My boot disk has an Exec file on it to transfer most of the stuff on the system disk (the Filer, Editor, Compiler, Assembler, Linker and other files). The system works considerably faster having these files (and program text files) on an Apple II Memory Expansion Card than on floppy disk, and makes Apple Pascal 1.3 a joy to use.

Keith J. Bernstein
The Bronx, N.Y.

After reading your letter, Dennis found the directions for Pascal Exec files in the Apple brochure "Addendum to the Apple Pascal Operating System Reference Manual." The brochure cautions that (as with DOS and Basic.system EXEC files) an unexpected occurence can throw the Exec commands out of sync with the prompts. For example, trying to Transfer a file to a RAM disk a second time may cause the prompt "Remove old <filename>?" to appear, which may send your Exec file off into the weeds.

Another consideration is the loss of the original SYSTEM.STARTUP file. Rather than replacing the original SYSTEM.STARTUP with the new file, you may want to rename the original and then "chain" to it at the end of the Exec file by adding a command sequence such as "X" (eXecute), and then ":OLDSU.CODE" (to reply to the prompt "Execute what file?") to the end of the Exec file.*

Incidentally, the "" prefix is used to tell Apple Pascal that the specified file is to be found (or created) on the boot diskette (see the Apple Pascal Operating System Manual for particulars).*

May I have your name?

The RWTS routine in the *DOSTalk Scrapbook* is great. I even use it to find volume names on ProDOS disks. How about a similar routine for ProDOS or even just a way to get the volume name? On page 3-17 of *Beneath Apple ProDOS* is a chart converting sectors to blocks. It indicates that sector 11 on track 0, which is where I find a disk's volume name, is the second half of block 6. But on page 4-8 it says that the volume name is in block 2. What gives?

Here's a tip you may be able to use in exchange for your help. You can load *any* ProDOS file into AppleWorks, not just text files. AppleWorks will change all control characters (ASCII 0 to 31 and 128 to 159) to the symbol "#". All other characters appear as themselves. This makes it easy to convert files from word processors that use binary files.

Guy Forsythe
Westerville, Ohio

Here's a couple of simple ways to find out a disk's name. Put the disk you're unsure about in a drive (slot 6, drive 1 for these examples) and type any of the following:

```
CAT,S6,D1  (volume name appears at top of catalog)

PREFIX,S6,D1   (set prefix to what's in S6,D1)
PREFIX         (ask for current prefix)

PREFIX /       (slash by itself voids prefix)
CAT,S6,D1      (make S6,D1 the active drive)
CAT
```

The final example is useful if you have several disks whose names you don't know. After the first one, all you have to do is insert the next disk and type CAT to learn the names.

*Gary Little has a simple Applesoft block read/write program in his book **Apple ProDOS: Advanced Features for Programmers**, pages 36-40.*

*The chart on page 3-17 of **Beneath Apple ProDOS** confuses just about everybody who tries to use it. See our July 1985 issue, page 55, for an explanation and for a **useful** ProDOS to DOS 3.3 sector conversion table.*

We tried your AppleWorks trick, and it works nicely.

Applesoft ONLINE call

While using ProDOS, I often have as many as four volumes on line at any one time. Out of necessity, I wrote a STARTUP program for my hard disk in which I included a subroutine to check each slot and drive and display any volume names (see listing below). This program can be run independently or you can make it a part of a larger program that needs to know what volumes are online. It's possible to put ProDOS volume names into an array for easy manipulation. I thought your readers might enjoy knowing how to "capture" a ProDOS volume name.

```
 5 REM *   VOLUME.FINDER    *
 6 REM *     S =slot        *
 7 REM *     D =drive       *
 8 REM *     V$=volume name *

10 HOME : S=1 : D=1 : D$=CHR$(4)
20 ONERR GOTO 50
30 PRINT D$;"PREFIX,S";S;",D";D
40 PRINT D$;"PREFIX" : INPUT "";V$
50 POKE 216,0 : CALL -3288 : REM fix ONERR
60 PRINT "SLOT ";S;" DRIVE ";D;" ";V$ : V$=""
70 S=S+1 : IF S=8 THEN D=D+1 : S=1
80 IF D<3 THEN 20
90 POKE 216,0 : REM clear ONERR flag
```

Joseph Kline
Lubbock, Texas

*Readers curious about line 50 should take a look at "Digging into DOS" in the January 1985 **Open-Apple**, page 2. Try adding these lines to your program:*

```
25 POKE 222,0
55 IF PEEK(222)=3 THEN 70
```

IIgs system utilities

We have had an Apple IIgs with 3.5 drives for several weeks now and it is super. But we don't like the long, interminable System disk boot up necessary to make backup data disks (it takes over 80 seconds just to get there). The QUICK COPY program itself is magnificent, but getting to it is the drag.

Is there any way, using the file moving options in the System Disk, to transfer a startup file and just the copy program to another disk, so that when we want to backup our 3.5 disks, we can have a separate, fast attack copy program? Or does such a program already exist on 3.5 disks?

Preston Boomer
Santa Cruz, Calif.

Interestingly, there are two complete system utility programs on the IIgs system disk. One is in the subdirectory called SYS.UTILS. This is an updated version of the IIc system utilities. It was written by Apple's own software engineers.

*The other is in the subdirectory called DESKTOP. It's a renamed version of **MouseDesk**, which was developed by a French company called VersionSoft and was once distributed in the U.S. by International Solutions, which has since gone out of business.*

Both of them are ProDOS 8 programs. Both of them can copy disks, but the disk-copy portion can't be separated from either. To make a disk that directly boots into one of these programs, start by initializing a new disk and copying the IIgs system disk program /SYSTEM.DISK/SYSTEM/P8 into the new disk's main directory. Then rename "P8" "PRODOS", for that is what it really is.

For Apple's system utilities, next copy all the files in /SYSTEM.DISK/SYS.UTILS into your new disk's main directory. That's it.

*For the **MouseDesk** look-alike, copy the files /SYSTEM.DISK/DESKTOP/DESKTOP1 and /SYSTEM.DISK/DESKTOP/DESKTOP2 into your new disk's main directory. Rename "DESKTOP1" "DESKTOP.SYSTEM". If you have a 3.5 disk, you could consider moving the other files from the DESKTOP directory to your new disk, too, but the program will start up faster without them. The other files won't fit on a floppy.*

*Besides the programs on the IIgs system disk, there are several stand-alone copy programs that have been designed to speedily copy both 5.25 and 3.5 disks. **ProSel** has one, as does Living Legend's Misk Disk #1 (see September 1986, page 2.64c). Bill Basham has already updated his **DiversiCopy** to use IIgs memory ($30, Diversified Software Industries, 34880 Bunker Hill, Farmington, MI 480018-2728, 313/553-9460).*

File lost, sort of

I've got my word processor data disk set up using very efficient subdirectories and I've had no problems. But I just saved a document as /DATA/LETTERS/APPLE and it's disappeared. This should have been the thirteenth entry in the directory but it only shows twelve. I know the file's there because I can use any other program and it shows in the catalog, but I can't find it with *Apple Writer* and I can't load it even though it's there.

Gee Uncle DOS, got any ideas?????

Michael C. Tiernan
North Reading, Mass.

You've been sorting directories with a buggy sort utility. See "Don't try to sort ProDOS directories" in our March 1986 issue, page 2.10.

*What you need is Glen Bredon's **ProSel.** A program called MR.FIXIT on that disk will fix your damaged directory. A program called CAT.DOCTOR will sort your directories correctly. You get an excellent program selector free ($40, 521 State Road, Princeton, NJ 08540).*

Booting ProDOS from DOS 3.3

What's the best way to print a path tree? I have a 20 meg hard drive and can get lost easily.

My hard drive is in slot 2. The kids play on the computer a lot, and if it is in 7 either 6 won't boot (hard drive off) or they turn the hard disk on and off.

When I boot my Sider with a PR#2, DOS 3.3 works fine, but I can't get into ProDOS without booting it elsewhere. I can't boot ProDOS from the Sider's menu.

P.S. The tip on Glen Bredon's *ProSel* is worth many years of your subscription price. The article on the IIgs is priceless. I spent $20 on magazines for a lot less information.

Bob Wilson

*The answer to your ProDOS path tree problem is easy if you already have **ProSel**; try running the program on that disk called INFO.DESK. And don't be embarrassed you didn't already know it was there — there's so much good stuff on that disk it takes*

weeks to find it all.

Likewise, getting ProDOS to boot from DOS is pretty simple. First, we have to get PRODOS into a BINary file that we can convert into a DOS 3.3 file (PRODOS is normally a SYStem file, which can't be converted). Do a CATALOG to find the length of the PRODOS file you want to convert. (PRODOS 1.1.1 is 14848 bytes long, PRODOS 1.2 is 15485 bytes.) Then:

```
BLOAD PRODOS,A$2000,TSYS
BSAVE PRODOS.OBJ,A$2000,L?????
```

*Now you can use CONVERT or **Copy II Plus** to transfer the new file PRODOS.OBJ to a DOS disk. To get the file onto the Sider, you'll have to convert it onto a 5.25 DOS 3.3 disk and then use a FID-type utility to get it to the Sider, since neither CONVERT nor **Copy II Plus** is equipped to deal with DOS 3.3 on devices other than 5.25 floppy drives. Once you have the file on the Sider, change its name from PRODOS.OBJ to just PRODOS.*

Now the only problem is that you can't BRUN the PRODOS file without a few considerations. After the PRODOS image is loaded and executed, it will relocate itself up into the top of the Apple memory. PRODOS by itself in memory is "all dressed up with no place to go," as it knows a lot about managing the disk drives but doesn't do much useful work by itself. PRODOS will therefore attempt to load a system program (such as BASIC.SYSTEM) to tell it what to do with the knowledge it has; PRODOS looks for the first system file in the "boot disk's" root directory that has a name ending in ".SYSTEM". If such a system file is not found, ProDOS throws up its hands and stops with an "UNABLE TO LOAD SYSTEM FILE" error.

This means that in addition to the DOS disk from which we are going to BRUN PRODOS, we also need a ProDOS volume on-line with a suitable system file so that ProDOS has something to load and run. And we have to be able to tell ProDOS where this disk is. It turns out that ProDOS assumes the boot drive's slot (times 16) is stored in zero-page location 67 ($43). Therefore, we need to POKE the desired slot number times 16 into that location before executing PRODOS. For your hard disk in slot 2, a program to start PRODOS would look like this:

```
10 REM *** Boot ProDOS from DOS ***
20 SLOT = 2 : REM slot number of ProDOS boot disk
30 PRINT CHR$(4);"BLOAD PRODOS,A$2000"
40 POKE 67,SLOT*16 : REM pass slot number
50 CALL 8192 : REM $2000; start ProDOS
```

After PRODOS starts up, it will now look in the ProDOS volume connected to slot 2, drive 1, for a system program. Make sure there's something there for it to execute and your problems should be solved.

VCR backup

Please expand on the subject of hard disk drive backup using a video cassette recorder. In the 'Apple IIgs Book' (Bantam) by DuPrau and Tyson, the authors comment on page 138 about using a device by Corvus and a VCR.

This would be significantly less expensive for someone that already owns a VCR than buying a backup device like B-Sider.

William D. Blessing
Tulsa, Okla.

Back in the Olde Days of microcomputers when hard disks were rare and tape backup units were almost non-existent, Corvus used to manufacture a product called the Mirror that allowed the use of a VCR as a back-up device for Corvus hard disk drives. We called Corvus's sales department (at 408-559-7000) to see if the device was still in production. We were informed that although Corvus still supports the unit, it is no longer being manufactured. The sales representative told us that some surplus units may be available.

Although we couldn't locate an old ad with the price, as we recall the unit cost several hundred dollars without the VCR, within the range of the B-Sider (which includes the tape drive unit). And the Corvus unit may not support non-Corvus drives since backing up a disk device is a matter of combined hardware and software.

No help available

Here's a short question you might be able to help me with. What can I do when I get a NO BUFFERS AVAILABLE error? This often happens when I have *GPLE* and *Double-Take* up and running in ProDOS. Could I be the only person with this problem?

John M. Sklar
Brown Deer, Wisc.

*The NO BUFFERS AVAILABLE message typically occurs when you are trying to open a file or otherwise access a disk and there isn't enough memory space available for a disk buffer (1K). You'll also get this message if you have the limit of eight buffers already open (rare), or if you attempt to BLOAD a file into a memory area that has been protected in the ProDOS system bit map (not rare). The protected areas are the zero page and stack ($0000-$01FF), the text screen ($0400-$07FF) and parts of Basic.system itself. **GPLE** and **Double-Take** may also set the map to protect the areas they are in.*

Here's a short program you can run to determine exactly what memory pages are marked as protected at any given time:

```
100 REM * ProDOS System Bit Map Reader *

110 PRINT "ProDOS System Bit Map" : PRINT
115 PRINT "   Pages marked X are protected" : PRINT
120 PRINT "  0 1 2 3 4 5 6 7 8 9 A B C D E F"
125 ADR=48984 : REM where bit map is in global page

130 FOR I=0 TO 11 : REM do 12 sets of 16 mem pages
140    I$=STR$(I)
150    IF I=10 THEN I$="A"
160    IF I=11 THEN I$="B"
170    PRINT I$;"-"; : REM print label at left edge
180    FOR K=0 TO 1  : REM two bytes=16 pages
190       V=PEEK(ADR) : ADR=ADR+1
200       FOR L=7 TO 0 STEP -1 : REM bit magic
210          V$=" "
220          IF V=>2^L THEN V$="X": V=V-2^L
230          PRINT V$;" ";
240       NEXT
250    NEXT
260    PRINT
270 NEXT
280 END
```

*The ProDOS versions of **GPLE** and **Double-Take** install beneath Basic.system and use up part of Applesoft's memory. If you load a long program, memory can start getting very tight. Execute a PRINT FRE(0) to determine how many bytes of free memory you have left at any particular time. If the answer is less than 1,024 or so, you simply don't have enough memory left to access the disk. The only alternatives are to split your Applesoft program into modules (see April 1986, page 2.17) or to remove **GPLE** or **Double-Take**. Note that **Double-Take** is modularized; you may be able to keep your favorite features of that program and remove only the features you don't often use.*

A better MouseText trap

I have a solution for those who desire to have the old MouseText-less character set on the IIc (November, page 2.78). The operation requires a hardware hacker's turn of mind, however.

What is involved is laying open the IIc and unsoldering the character generator chip (Video ROM) from the motherboard. The squeamish need not apply as there is potential for disaster here. The chip in question is located under the disk drive. The clearance is such that it is unlikely that a chip socket can be installed here for a new chip.

The new chip to be installed is a 2764 EPROM (approximately $4) into which you must program a copy of the old character generator data from a borrowed pre-enhancement IIe. This data goes into (let's arbitrarily say) the lower half of the 2764 and the IIc's present data goes into the other half of the 2764.

You'll also need to make a hole through the IIc's case for access to a miniature toggle or slide switch. A conveniently unused cranny is available just above the 'shelf' near the speaker volume control. A generous blob of epoxy will serve to anchor the switch in place.

A few things to note: The character generator ROM data is not copyrighted. Every time you trade in your IIc motherboard for the newest upgrade you have to make this modification all over again. Worst of all, your dealer may not accept a modified motherboard on an exchange basis, which could put your ability to get upgrades in jeopardy. And since broken IIcs are usually fixed by means of a motherboard exchange, you may even have trouble getting a modified IIc repaired without unmodifying it first. Save those old parts!

The IIe and IIc character generator ROMs are equivalent to 2732 EPROMS and are 24 pin devices. The 2764 is a 28-pin chip and contains twice the number of locations as the 2732. The early IIc motherboard that I worked with was laid out to accommodate 28 pin chips. Could Apple have been on the verge of installing a 'dual' character generator? If not, why not?

You must bend pin 2 out to the side before soldering the 2764 to the motherboard. Pin 2 will not be soldered to the motherboard. The select switch controls whether pin 2 is yanked low or pulled high, thus determining which character set is displayed. A pull-up resistor, perhaps 4.7K ohms, must be soldered between pin 28 and pin 2. Then pin 2 is wired to one terminal of the selector switch. The other terminal of the switch is connected to a convenient ground. Such a ground connection can be made by securing the bared end of the wire under the motherboard mounting screw at the left front corner.

You can flip the switch at any time (powered up or not) to make MouseText go away. Also you can switch MouseText back on just as conveniently. The software doesn't (and won't) need to know what is going on.

This same scheme, using exactly the same chip, applies in principle to IIe's. The practice is slightly different in that the 2764 is inserted into a 24 pin socket with pins 1, 2, 27 & 28 hanging off the keyboard end of the socket. Again, pin 2 is manually controlled by a switch and pulled high through a resistor. Also, solder a jumper to connect pins 26, 27, 28, and 1. This provides 5 volt power from the socket's power hole to the chip's power pin and makes the other mentioned pins happy.

Note also that the concept can be applied to

produce ROMs containing switchable English and Spanish (or other) character sets. Also, a 27128 or larger ROM could be employed to produce a killer character generator. All that is needed is a suitable multiple-position switch and the right connections to map in the appropriate 4K chunk.

Those not wanting to go to the trouble of burning their own EPROM may want to check out a product from Bone Frontier in the November 1986 *inCider*, page 158. I haven't seen their gizmo but it sounds just like what I've described here for the IIe. They don't mention the IIc. It's $50 price seems a bit steep, however, when you consider that's 5/6ths of the cost of an EPROM programmer board from JDR-Micro.

Incidentally, the EPROM burner I used for this exercise reports a checksum of $F800 when reading a standard IIe character generator ROM. The checksum of an enhanced IIe character generator is $B595. Appropriately enough, combining these two chunks of data into a single 2764 yields a ROM whose checksum is reported as $AD95 ($F800 + $B595 = $AD95, with a carried bit lying around on the floor).

As an aside, I had originally offered to perform this modification for a friend, free of charge except for a few bucks to cover the cost of parts. After I told him the details, he had second thoughts and changed his mind about the proposed surgery. I then offered the same deal to another friend. The second friend accepted the offer and is still very pleased with the outcome. He also allowed me to take his modified IIc to my user's group where I demonstrated it and offered my services to the group to perform the mod. The interest was underwhelming. Here is an idea every bit as good as sliced bread, and yet I feel it will take a marketing maven and not a marketing Marvin to make it fly. Maybe the $50 is not so out of line after all.

Marv Wager
San Diego, Calif.

Open-Apple

[redacted]

is written, edited, published, and

Business Consultant Richard Barger
Technical Consultant Dennis Doms
Circulation Manager Sally Tally
Business Manager Sally Dwyer

Open-Apple
P.O. Box 7651
Overland Park, Kansas 66207 U.S.A.

Published monthly since January 1985. World-wide prices (in U.S. dollars; airmail delivery included at no additional charge): $24 for 1 year; $44 for 2 years; $60 for 3 years. All single back issues are currently available for $2 each; bound, indexed editions of Volume 1 and 2 are $14.95 each

Open-Apple
P.O. Box 6331
Syracuse, N.Y. 13217 U.S.A.

Subscribers in Australia and New Zealand should send subscription correspondence to ***Open-Apple***, c/o Cybernetic Research Ltd, 576 Malvern Road, Prahran, VIC 3181, AUSTRALIA.

Open-Apple is available on disk from Speech Enterprises, P.O. Box 7986, Houston, Texas 77270 (713-461-1666).

ISSN 0885-4017 **Source Mail: TCF238**
Printed in the U.S.A. **CompuServe: 70120,202**

Slot for terminal needed

I need to know if it's possible to use my Super Serial Card to drive both my printer and a modem. While I realize that it can't be done concurrently, is there a way I can buy (or rig) a switch to do it? I have a fair understanding of wiring, and I realize that the jumper block on the SSC is a modem eliminator. To my understanding, this simply reverses some of the lines, and it doesn't seem it would be so tough to have an external switch that could flip between either the modem or the printer, and reverse the lines depending on which peripheral was being addressed. So, can it be done? How?

My reason for asking is, of course, because I'm out of slots. My printer is in slot 1, a clock card in 2, an accelerator in 3, a mouse in 4, two UniDisk 3.5's in 5, a DuoDisk in 6, and a RAMFactor in 7. Help!

A possible solution I've read about is SRCG's Switch-a-Slot, but I suspect that to use it I would have to turn off my computer to switch slots. This would be so inconvenient with my hardware/software setup that it wouldn't be worth the effort. Of course, I may be wrong; the advertising doesn't address it, and as usual my dealer is as useful as an extra nostril. Do you know if I can use the Switch-a-Slot without turning the computer off?

Finally, I need to know if I can get software to emulate an IBM terminal connected to an IBM System 36 in our building. This, of course, is why I want to hook up a modem in the first place. Actually, I have to manage it, or my department may take away my beloved IIe and give me (ick!) an IBM PC.

Finally, is there an accelerator card for the Apple III, or an accelerator for the IIe that can be modified to work with the III?

Dean Esmay
Flossmoor, Ill.

RS-232 switches are common and available from a number of sources (at widely varying prices). For example, Moore Business Forms has a number of different switches in its computer products catalog, ranging in price from $80 for a simple A-B switch to $220 for a box that can switch between six devices. You would also need a ***null modem*** *cable. If your Super Serial Card is set to "printer," use the null modem cable between the switch box and the modem. Conversely, if the SSC is set to "modem," use the null modem cable between the switch box and the printer. If you decide to build the switch box yourself, don't attempt to put any crossovers in the switch itself—just run everything straight through.*

RS-232 wiring can get extremely complicated in no time at all. If a simple null modem cable doesn't seem to work, you might want to look at "You Can Make the Right Connections", in the June 1986 ***A+****, pages 53-64.*

Southern California Research Group's Switch-a-Slot does require you to power down before changing the selected card. Another product supposedly available is called ProSlot (from CGL Microsystems, 10407 Blue Ridge Blvd., Suite 522, Kansas City, MO 64134, 816-383-6619). Dennis has talked to the company's answering service several times but has been unable to get anyone who knows anything about the product to call him back. According to an ad in the February 1986 ***Nibble****, page 59, the device fits into slot 7 and provides two software-selectable slots; it costs $119.95 plus $4 shipping and handling.*

Dennis says that terminal emulation for the IBM is tricky if your system 36 is close-mindedly using the IBM "standards" for communications and character codes. IBM only recently decided to recognize ASCII devices; many older IBM systems use a different byte-to-character code known as EBCDIC. Hooking up to such a machines often requires that a rather expensive emulation device be connected to your computer to make it look like an IBM terminal.

Most current IBM systems will allow you to define your terminal as an ASCII device and will communicate with you through a device called a protocol converter (not the same as the Apple Smartport gadget discussed at length last month). You need to check with your data processing people and see what kinds of terminals they will support, then find a software package that will emulate that device. ***ASCII Express, the Professional****, for example, will emulate an IBM (ASCII) 3101 terminal.*

One thing you have to watch out for is that some "emulations" do not fully duplicate the function of the original terminal for some hardware-specific items such as special function keys; for example, ***AE Pro's*** *emulation of the DEC VT-52 does not include generation of the keypad function key codes. Softronics'* ***Softerm 2****, which performs 24 terminal emulations (including an IBM 3101-20 in block mode), includes a hardware device to act as an ancillary keypad for some emulation functions.* ***Softerm 2*** *is $195; you can reach Softronics at 303-593-9540.*

We don't have an answer to the "accelerator card for the Apple III" question. The best answers to Apple III questions come from the folks at ***On-Three****, P.O. Box 3825, Ventura, CA 93006, 805-644-3514.*

Communications software

Do you know of a good (and inexpensive) communications package that runs under ProDOS? I have been using Hayes Smartcom I that comes with my MicroModem IIe, but it captures files under DOS 3.3. I have to convert them to ProDOS before using them. Also, I cannot install Smartcom on my UniDisk 3.5.

Joe Movich
Alta Loma, Calif.

There are a large number of ProDOS communications packages. Most of them have been included in comparative reviews in Apple magazines recently. Go to your local library and look at "Telecommunications Software" in the April 1986 ***A+****, pages 24-36 (14 packages compared); "Telecommunications Software Review" in the January 1987* ***Call-A.P.P.L.E.****, pages 11-19 (five packages compared); and "Telecommunications, The Software Connection" in the February 1987* ***inCider****, pages 53-62 (nine packages compared).*

In addition, some word processors, such as ***Apple Writer****, and most desktop accessory programs, such as* ***Pinpoint****,* ***Jeeves****, and* ***Fingertips****, allow rudimentary use of a modem and transfer of text data.*

There is also a public domain terminal program, called ***Kermit****, that is widely used in the university environment. If you are associated with a college or university, your data processing people should be able to get you a copy of the ProDOS version of this program.*